2..Documentation Document:

5-Joined Book and Loan tables to calculate how often each book is borrowed.

JOIN Loan l ON b.BookID = l.BookID

Used a LEFT JOIN with Review so books

LEFT JOIN Review r ON b.BookID = r.BookID

Applied aggregate functions:

Count(l.LoanID) AS TotalLoans,

AVG(r.Rating) AS AvgRating

Used `HAVING COUNT(l.LoanID) >= 3`

It combines usage data (loans) with user sentiment (ratings) to give a balanced

6- JOIN Loan l ON m.MemberID = l.MemberID

JOIN Book b ON l.BookID = b.BookID

LEFT JOIN Review r ON m.MemberID

ORDER BY m.FullName, l.LoanDate

Ordered by member name and loan date

Query 7

- **Joined Book and Loan .**
- **Used a LEFT JOIN with Payment so loans without fines are still counted.**
- **Applied `ISNULL(p.Amount, 0)`**

`Example: sp_IssueBook`

- **Prevents issuing a book that is not available**
- **Stops issuing if the member already has maximum allowed loans**
- **Ensures book status updates correctly.**

Any assumptions you made

- **Members can borrow books from any library**
- **Not every loan results in a fine**
- **Reviews are optional for members**

Testing Evidence

Successful Execution

```
EXEC sp_IssueBook
    @MemberID = 1,
    @BookID = 5,
    @DueDate = '2025-01-15';
```

- Procedure executed successfully
- Book issued to member
- No error message displayed

Book Not Available (Error Handling)

- Procedure stops execution
- Error message displayed